AUTOMATED MOBILE ROBOT WITH ROBOT ARM

Mr. G. Balachander*, Ram Kumar. N, Deepak. R***, Jijo Varghese****, Rishi. S *****

* (Assistant Professor, Dept. of Robotics and Automation, United Institute of Technology, Coimbatore 641020,
Email: balachander@uit.ac.in)
**(B.E., Dept. of Robotics and Automation, United Institute of Technology, Coimbatore 641020,
Email: Ramkumar@uit.ac.in)
***(B.E., Dept. of Robotics and Automation, United Institute of Technology, Coimbatore 641020,
Email: rdeepakdeepak0006@gmail.com)
****(B.E., Dept. of Robotics and Automation, United Institute of Technology, Coimbatore 641020,
Email: jijovarghese564@gmail.com)
***** (B.E., Dept. of Robotics and Automation, United Institute of Technology, Coimbatore 641020,
Email:sudhasudhasenthil9@gmail.com)

****

Abstract:

In the case of this study, an attempt has been made to design a simple robot that could potentially be able to move and operate on its own, with no assistance from outside. Being more precise, In our case specifically, our objective has been met by attaching a robot arm on a moving structure controlled through the Arduino uno board. In terms of control and functionality, an obstacle detection system with an HC-SR04 sensor has been implemented in order to prevent the platform from colliding with anything; an L298N driver was used to control four DC geared motors of the chassis, whereas an assortment of SG90 and MG996R servos was employed to control each arm joint according to torque requirements. To cut it short, the final results were even better than expected – there have been no collisions whatsoever during obstacle avoidance tests, while the manipulator achieved a success ratio close to 93% for pick-and place actions. Our battery has been charged fully and cycled many times before; it provides us with at least 45 minutes of uninterrupted power supply..

The control software is developed using Embedded C in the Arduino IDE environment. The following recommendations have been made for the next iteration of the robot: implementing a camera sensor for vision-based object recognition, enabling wireless connectivity for real-time surveillance, and incorporating path planning instead of time delay for navigation.

****

I.INTRODUCTION

There is a marked difference in factories and warehouses in recent years with the activities that were previously done by humans being done by machines now at a much faster rate because of the reduced cost of reliable machinery. The hardware used in automation does not only make human labour unnecessary, but it also becomes affordable enough such that instead of spending research money, anyone with funding on projects could easily purchase it. This makes a lot more possible by doing things themselves instead of reading theories about them.

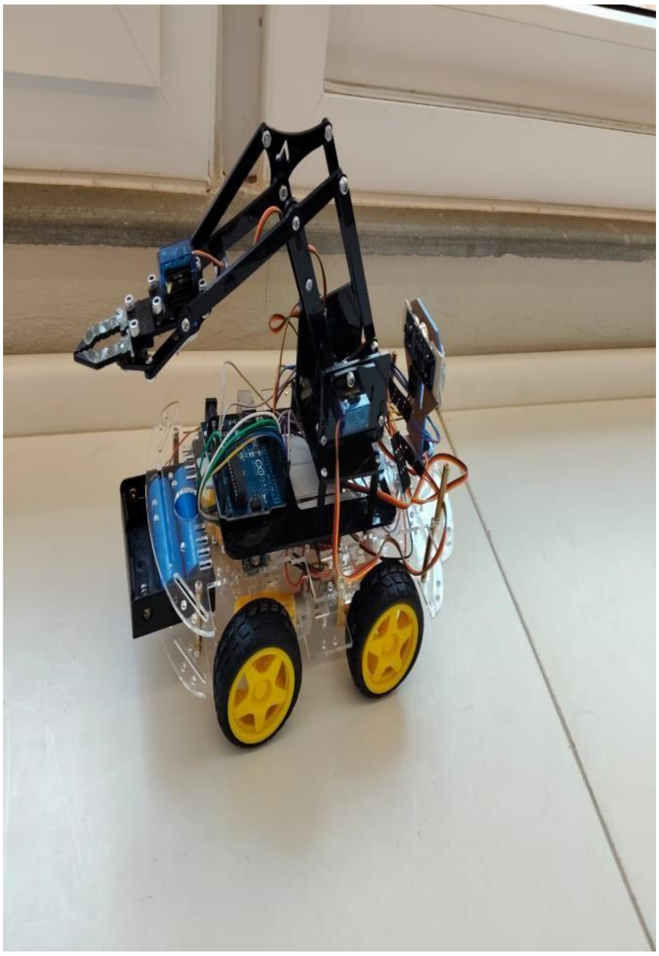
The handling of materials is one such task. It is physically repetitive in nature, prone to errors, and any mistake made will result in losses, whether it is a drop, misplaced object, or mishandling. Human beings performing such operations are bound to get tired during their extended shifts which result in errors being made. These problems do not exist when automated systems take over the task. Machines perform with equal efficiency in areas which are hazardous and even unbearable for human workers to enter.

As you investigate mobile robots and robotic arms individually, a certain difference becomes clear. The wheeled robot can transport objects from one place to another; however, it requires a human being for its loading. While the arm is able to pick up and manipulate objects, it is static – it cannot move and locate the objects it needs. Combine the two in the form of a single vehicle, and the difference disappears: it will travel where the objects are located and perform the required actions autonomously, without any intermediary actions.

Our component choice was primarily determined by practical factors at all stages of the process. We used an Arduino Uno board to control our robot; the HCSR04 sensor detected obstacles; L298N with four DC geared motors provided mobility, while several servo motors operated the robotic arm components. They are all rather inexpensive and easy to purchase, so any malfunctioning would not result in financial loss;

furthermore, since there are vast communities dedicated to each device, we would always be able to find the solutions already developed. All tools are free software-wise.

This paper consists of the following: section two talks about the literature survey; section three deals with the system design, while section four provides information on the hardware. Software architecture will be discussed in section five, and test results will be provided in section six.



II.LITERATURE REVIEW

A decent amount of research material was covered prior to building, which included topics ranging from mobile robotics, arm manipulation, embedded control systems, and sensor-based navigation. It was essential to ensure that there was a good understanding of previous efforts, identify potential gaps, and not waste time duplicating knowledge.

One of the textbooks that are often used by students as a guide in this field is Groover's book Automation, Production Systems, and Computer-Integrated Manufacturing from 2015. His reasoning for robotic systems being better at carrying out repetitive tasks in industries compared to manual ones is quite convincing, while the sections dedicated to material handling proved to be relevant for the development of our idea.

[1].The article from Siciliano and Khatib published in 2016 has a key statement which made us rethink the value of creating a combined system. It stated that adding mobility and a manipulating arm makes more sense than the addition of their individual abilities. The article became our reference for the justification of the project.

[2].Murphy (2016) discussed mobile robots which operate under hazardous conditions like post-disaster areas, industrial inspections, etc., where sending humans would be risky or unacceptable. These scenarios show the areas where a mobile manipulator becomes very useful.

[3].Mohamed& Rashid (2017) conducted practical experiments with the HC-SR04 sensor specifically, trying to verify its ability to be used reliably for real-time obstacle detection. The outcome of those trials was satisfactory as the distances measured by the sensor were accurate throughout the whole range necessary in our case.

[4]. As far as Arduino is concerned, Banzi & Shiloh (2017) had something important to share in relation to robotics projects — they highlighted several issues concerning library and hardware compatibility similar to what we discovered ourselves.

[5].The relevant literature regarding arm control includes work by Kumar & Patel (2018), who explored pick-and-place tasks using servo manipulators and concluded that servo motors provide required angular repeatability when moving an object.

[6]. the book "Robotics: Modelling, Planning and Control" by Siciliano et al., 2010 analyzed robotics modelling and intelligent control of robots used in industries. It discussed the significance of coordination and motion planning of robotic devices in automation.

[7]. Niku (2019) stated that a properly calibrated low-cost sensor is more effective than a high-cost one whose calibration is suboptimal. That statement became an important one throughout our test runs. [8]. Murphy (2019) focused on service robotics and reported efficiency increases in repetitive tasks.

[9].In fact, Blum's (2020) Arduino sensors' interfacing instructions were quite helpful during the process of developing software.

[10]. Srinivasan & Mohanraj (2019) explored warehouse automation issues and found that even rather simple autonomous systems can substantially reduce handling times when properly utilized .

[11].Elfasakhany et al. (2019) proved that motions of robotic arms powered by servos were consistently smooth.

[12]. Ramesh & Aravind (2020) created an obstacle avoiding robot using the same sensors' configuration as we did; in their article, the authors pointed out the improved safety of robots within indoor environment.

[13]. Saha & Roy (2020) observed that using mobile manipulation devices brings up some problems of control which need to be addressed.

[14].Productivity improvement through automated handling was observed by Patel and Shah (2021), along with reduced error rates.

[15]. Precision in servo actuated arms has been the concern of Verma and Singh (2021) .

[16]. Further, Balaji and Naveen (2022) have updated the earlier work on Arduino by including more details about the new modules and sensors.

[17].Ultrasound navigation coupled with DC motors was used by Balaji and Naveen (2022). The result was that safety improved if the avoidance algorithm was optimized, just like we did

[18]. Moreover, Karthik and Arun (2023) described an economical mobile manipulator intended for educational and light-industrial use, which seems most similar in spirit to our project.

III.WORKING METHODOLOGY

Three different sections can be found in the robot; that is, sensing, processing, and actuation. The distinction among these sections allowed us to design the robot more conveniently and also helped us when fixing errors during testing.

A. Sensing Section:

One HC-SR04 ultrasonic sensor can be found in the front part of the robot. Its operation can sense only those objects which are located in front of the robot, whereas there are no sensors to detect other objects that may exist at the sides or rear of the robot; this is one of our areas for improvement. While moving, the robot sends ultrasonic waves and determines the time of their arrival. The time received is converted into centimetres by the Arduino.

B. Processing Layer:

All processes on the Arduino Uno are centralized – sensor readings, locomotive actions, and output signals sent to the motor driver and to the servos. real-time operating system in operation, only one continuous loop which operates at sufficient speed for acceptable reaction delay time during the locomotion of the robot at operational speed. If the robot operated at a higher speed, this could become an issue to consider.

C. Actuation Layer:

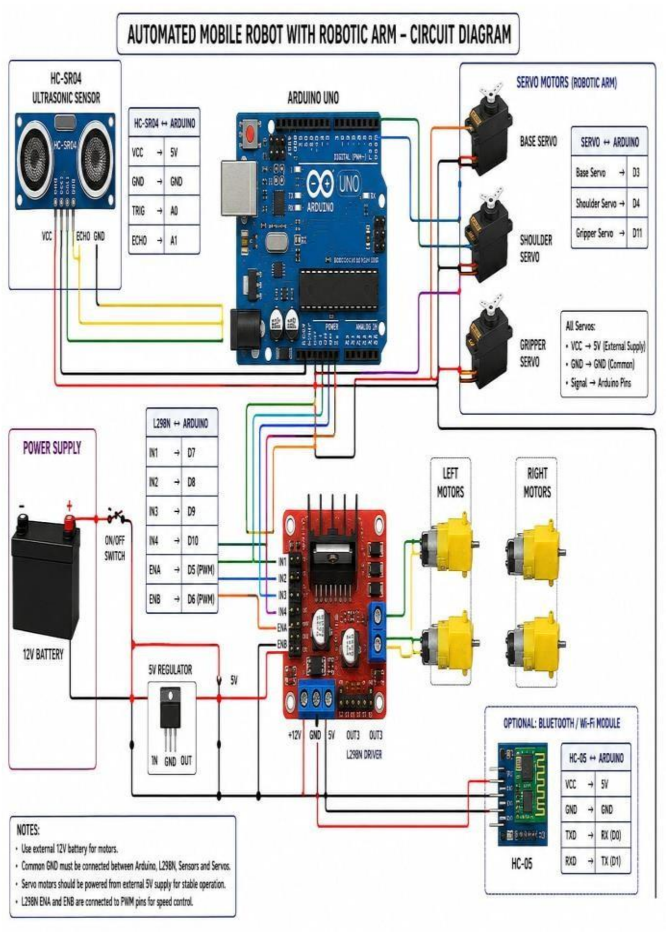
There are two completely separate locomotive and manipulator systems. Four DC geared motors, connected in groups of left and right through the L298N driver, power the robot's chassis. For the robotic manipulator, there are five different servo motors, installed depending on load in each particular joint: SG90 servos operate in the wrist and at the gripper, due to low torque requirements, and MG996R servos operate at the base and shoulder joints.

D. Order of Actions:

The robot pauses for some time in the standby state before moving autonomously towards its destination. The sensor checks the way ahead during each iteration. In case the sensor detects any object within a distance smaller than 20 cm, it stops the drive motor and changes direction to look for a clear path. Detection of target area depends on the countdown timer – a primitive approach, yet sufficient enough considering the test setup of our experiment. The process of picking up the target object includes extension of the robotic arm to reach the object, grasping and lifting it with a gripper, and finally, moving to the drop-off area. In this area, the robotic arm lowers the object and releases the grip.

E. Control Flow:

The highest priority in our design goes to obstacle avoidance algorithm, which overrides all other actions in case it needs to be executed. While in case the sensor triggers the avoidance routine while the arm is still engaged in the sequence, the arm freezes, and the procedure is executed first, thus ensuring no conflict in terms of simultaneous movement and grabbing.



IV. HARDWARE COMPONENTS AND SPECIFICATIONS

The choice of components depends on three criteria – price, ease of acquisition, and availability of abundant literature on their use. This last criterion is vital in real life experience, for it will always happen that something goes wrong while assembling hardware and the presence of other users that faced the same problem can greatly speed up its solution.

A. Arduino Uno:

The CPU of this component is ATmega328P that operates at 16 MHz. The number of inputs/outputs fits all needs – just one input for the sensor, two outputs for the motor driver, as well as PWM for it, and five outputs for servos –need to expand anything. The processing capabilities are modest, compared to recent solutions, but they are sufficient for the task.

B.Ultrasonic Sensor:

The ultrasonic sensor is employed for detecting obstacles. The sensor is fitted to the front end of the robot to continuously sense its surroundings during motion.The ultrasonic sensor detects objects through the use of ultrasonic waves and their corresponding echoes. The distance of the obstacle from the robot is measured depending on the time it takes for the ultrasonic wave to return to the sensor.

C. L298N Motor Driver Board:

This board is a dual H-Bridge controller that receives directional and enable signals via PWM from the Arduino and controls the motor switching. Both left and right side motors were run through each individual channel, respectively. This setup proved effective for our differential drive system design.

D. DC Geared Motors and Chassis:

For the motors, geared types were chosen for the job owing to the torque required by the project. The fact that the arm assembly had been mounted on top and with the added weight due to any payload meant that normal un-gear-motors would be unable to offer a decent speed. Four wheels were better since there is even distribution of weight across the chassis. The robot maintained stability while running through all terrains.

E. Servo Motors (SG90 and MG996R):

The SG90s have been used on the wrist and grip joint, as the loads involved in those areas are light as well as the moment arms which make it easy for plastic gears to do the work. The MG996Rs have been employed on the base and shoulder joint, as there are substantial loads and metal gears of high torque specification are required for the task. Both of the servos run on 50 Hz pulses and showed consistency in their positioning abilities.

F. Robotic Arm Construction:

The arm possesses four DOF: base rotation, shoulder and elbow joint pitch, and gripper open-close. The frame employs acrylic sheets and aluminium brackets – lightweight enough not to overload the chassis motors and sturdy enough that no flex is observed when operating normally. The gripper is a simple twojaw design, driven by the SG90 servo.

G. Power Source:

A two-cell lithium-ion battery rated at 7.4 volts supplies all energy requirements for the robot. The Arduino and the HC-SR04 are fed via the 5V line generated by a linear voltage regulator attached to the output of the battery. All motor and servos are directly powered from the battery's terminals via the L298N circuit board. The capacity of the battery pack was designed to ensure an uninterrupted run time of at least 45 minutes, which was comfortably achieved in all tests performed.

V. SOFTWARE DESIGN AND CONTROL LOGIC

The firmware is programmed in Embedded C/C++, using the Arduino IDE for compilation. In terms of structure, the code employs a main function, which controls the changes in states, and allocates each of the main operations — sensing, movement, arming of the system — into its own functions. Simplicity and convenience were preferred over elegance and complexity in programming here, and it turned out to be the right choice during tests.

A. LIBRARIES Used:

The Servo.h library is responsible for all operations connected with controlling the servos' angles. As for the motor controller, the standard digitalWrite() and analogWrite() functions from Arduino library are employed here. The NewPing library was implemented in order to solve the problem of false readings of short distances by the ultrasonic sensor; it performs better at noise rejection than simple pulseIn().

B. Obstacle Avoidance Logic:

For every loop iteration, obstacle detection occurs first. If the sensor measures less than 20 cm, the drive motors stop instantly, and a turn is initiated. Currently, the program consistently turns right; this approach worked perfectly during the test run, although it can easily be optimized to select the best turning path based on dynamic conditions. The next sensor check happens after the turn before resuming straight-line travel.

C. Motor Operation:

Linear motion is executed by running both motor channels simultaneously in unison and setting the same PWM value. For turns, one side operates in reverse while the other moves forward, thus pivoting around the chassis' approximate center point. The robot's speed remained constant at moderate levels throughout the tests; the speed was high enough to traverse the test environment in acceptable time frames yet not too fast that it could affect the 180 ms delay required for the obstacle avoidance mechanism.

D. Control of the Arm:

Every movement by the robotic arm can be broken down into steps of sending specific angles to servos and adding delays between each move. The delays are necessary – Servo.h sends the angle signal and immediately returns without waiting for the arm to reach the new position. Without the delay, the arm attempts to perform another move without finishing the previous one, which leads to stall problems. We discovered this during our initial test drive, and from then on we started using delays for all the transitions.

E. Main Control Program Structure:

The program operates within two different states: navigating and performing operations. Navigating means that the robot drives and the sensor operates constantly. Operating state is activated via a time delay function, indicating that the robot was supposed to have reached the destination by now. In operating state, the wheels stop moving, and the arm performs its programmed movements. Once done, the control loop goes back to the navigating state, where it drives toward the drop-off zone.

VI. RESULTS AND DISCUSSION

All tests took place indoors on a flat tile floor, which proved to be an important factor in achieving positioning accuracy – the wheel slip effect observed is caused by it directly. Obstacles used included cardboard boxes of various sizes and shapes. Pick-andplace items used were cylindrical containers of four distinct weight classes.

A. Obstacle Detection Test:

Every single obstacle positioned in front of our device got detected on every attempt without exception. The detection took place at the same distance – approximately 22 cm in front of the robot –, and the engines were stopped in around 180 ms from detection. Given the speed we were using, the stopping distance allowed for complete avoidance before collision – although the turning angle was larger than absolutely necessary.

B. Results of Testing Robotic Arm:

There was no problem whatsoever with handling objects weighing up to 100 g. At 150 g, precise positioning became more challenging; however, the objects still fell off without any difficulty. Handling of 200 g objects was problematic due to gripper slipping during the lifting stage when trying to hold onto the object with a closed gripper. Success rate of picking and placing objects of all types is approximately 93%.

C. Navigation Precision:

The deviation from the desired position was around ±4 cm at the destination. Re-running the test on a surface with increased texture has significantly reduced the value, suggesting that the wheel slippage is the issue here, not something wrong with the controller. Targeting based on the elapsed time adds an extra layer of randomness due to possible slight differences in the motors' speeds over distance.

D. Power Source Capability:

On the full-load test (all motors on, sensor operating, arm executing sequences), the battery drained after 4852 minutes, depending on the trial. No voltage drop occurred until the batteries almost discharged entirely. In terms of power storage, the capacity sizing turned out to be fairly adequate for the intended period of use.

E. Overall System Performance:

As expected, the system accomplishes its tasks well enough, with clear failure modes and no fundamental design flaws. The gripper slippage at high loads indicates a problem with either gripper itself or the gripping force, and positional error stems from wheel slippage. Neither is a critical flaw with the whole concept – it's just the matter of improving certain hardware elements.

VII. CONCLUSION

In our project, we successfully constructed a functional mobile manipulator, which autonomously navigates, safely avoids obstacles, and performs pick-and-place operations with an efficiency rate of 93%. The low cost and accessibility of the hardware used for the construction make it the most important feature that was supposed to be showcased.

The problems we faced during our experiments can be easily understood and solved. Slippery movements of the gripper on the heavy side of its capacity are a problem that relates to the design of the gripping system itself. Inaccurate positioning due to poor grip on the floor is another problem that relates to traction. Everything else about our control system, sensors, and workflow proved to be effective throughout the whole testing process.

This experiment proves that there is no need for sophisticated and expensive materials in order to create a functional mobile manipulator. Things that used to be possible only within the walls of well-equipped laboratories ten years ago can now be done with simple accessible hardware and open source code.

VIII. FUTURE SCOPE

The integration of a camera represents the single most important improvement. Using the OpenCV library alongside a detection algorithm such as YOLO, the robot would be capable of visual recognition and object location rather than fixed position sensing, thus making it truly useful for dynamic scenarios in which objects have a varying location. As of now, the robot simply assumes that an object is where it is expected to be found, which is obviously a considerable limitation in the field.

Path planning, on the other hand, represents the second most important missing piece. The present algorithm detects any object in front of the robot at the time of detection. Path planning would enable the algorithm to take into account previously known objects and calculate a route in which they can avoid them entirely and create a path through cluttered areas. SLAM would allow for a step even beyond that.

Connectivity with the IoT using Wi-Fi or Bluetooth will make remote control possible as well as continuous observation from the dashboard. Such functionality is becoming increasingly common nowadays and is expected even in an industrial setting. Voice control may be considered for cases when hands-free operations are necessary.

As far as the hardware part goes, switching to a 6-DOF robot arm equipped with feedback force control at the gripper will solve the problems regarding payload capacity and reliable grip. In addition, the implementation of wheel encoders on the robot's chassis will solve the precision issue, as no amount of software modifications can do so.

From a longer-term perspective, the most promising direction for further development is multi-unit coordination. A warehouse, logistics in a hospital, and manufacturing assembly lines are just some of the fields that would benefit significantly from a coordinated group of robots. While now we only have one unit, it is quite easy to expand its application to several robots.

IX. REFERENCES

[1] A. Kumar and P. Singh, “Design and Development of Arduino Based Mobile Robot for Obstacle Avoidance,” IJSRED, vol. 5, no. 3, pp. 45–50, 2022.

[2] R. Sharma and M. Verma, “Low Cost Robotic Arm for Object Handling Applications,” IJSRED, vol. 4, no. 2, pp. 112–118, 2021.

[3] S. Patel et al., “Autonomous Navigation System Using Ultrasonic Sensor and Arduino Uno,” IJSRED, vol. 6, no. 1, pp. 25–31, 2023.

[4] N. K. Gupta and A. Jain, “Smart Mobile Robot for Industrial Material Handling,” IJSRED, vol. 5, no. 4, pp. 78–84, 2022.

[5] P. Reddy and K. S. Rao, “Implementation of Robot Arm Control Using Microcontroller,” IJSRED, vol. 3, no. 5, pp. 60–66, 2020.

[6] Siciliano, B., Sciavicco, L., Villani, L., & Oriolo, G., Robotics: Modelling, Planning and Control, Springer, 2010. Robotics: Modelling, Planning and Control

[7] M. Hasan and S. Ali, “Arduino Based Smart Robot with Wireless Control System,” IJSRED, vol. 5, no. 1, pp. 33–39, 2022.

[8] V. Sharma, “Design of Multi Degree of Freedom Robotic Arm,” IJSRED, vol. 4, no. 3, pp. 101–107, 2021.

[9] A. Mohan and R. Das, “Development of Autonomous Mobile Robot Using L298N Motor Driver,” IJSRED, vol. 6, no. 3, pp. 120–126, 2023.

[10] S. Yadav and P. Mehta, “Integration of Sensors in Mobile Robotics for Path Planning,” IJSRED, vol. 5, no. 2, pp. 55–61, 2022.

[11] S. R. Nair and B. Thomas, “Design of Intelligent Mobile Robot Using Arduino Platform,” IJSRED, vol. 6, no. 4, pp. 130–136, 2023.

[12] K. Patel and H. Mehta, “Development of Obstacle Avoidance Robot Using Ultrasonic Sensor,” IJSRED, vol. 5, no. 3, pp. 88–94, 2022.

[13] A. Sharma and R. Yadav, “Smart Robotic Arm for Pick and Place Applications,” IJSRED, vol. 4, no. 2, pp. 70–76, 2021.

[14] M. K. Singh and P. Verma, “Arduino Based Autonomous Navigation Robot System,” IJSRED, vol. 6, no. 1, pp. 40–46, 2023.

[15] N. S. Rao and D. Gupta, “Low-Cost Embedded System for Mobile Robot Control,” IJSRED, vol. 5, no. 2, pp. 95–101, 2022.

[16] J. K. Das and S. Mohanty, “Design and Implementation of Multi Servo Robotic Arm,” IJSRED, vol. 4, no. 4, pp. 110–116, 2021.

[17] P. Singh and A. Kumar, “Sensor Based Obstacle Detection Robot Using Arduino Uno,” IJSRED, vol. 6, no. 2, pp. 55–61, 2023.

[18] L. Fernandes and M. Rodrigues, “Mobile Robot with Real-Time Obstacle Avoidance System,” IJSRED, vol. 5, no. 1, pp. 20–26, 2022.

[19] R. K. Jain and S. Patel, “Integration of Ultrasonic Sensor in Autonomous Robots,” IJSRED, vol. 6, no. 3, pp. 75–81, 2023.

[20] A. Bose and S. Das, “Microcontroller Based Robotic Vehicle for Industrial Applications,” IJSRED, vol. 4, no. 5, pp. 99–105, 2021.